

---

**Amendments to the Specification:**

Please replace paragraph [0050] of the published application with the following corresponding amended paragraph:

**[0050] Matlab Simulation Listing**

```
% viterbi demodulator for phase excursion
%

% number of state transitions
K = 2000;

% number of states
M = 30;

% phase angle of each state
state_angle = (2*quadraturepi/M) * [0:M-1]';

% model the actual signal phase
phi_sig = .03 * [1:K]';

% actual signal
SNRdB = -6;
```

---

```
SNR = 10^(0.1*SNRdB);

u = exp(1j*phi_sig);

x = randn(K,1) + 1j*randn(K,1) + sqrt(SNR) * u;

phi_meas = angle(x);


% metric for phase state

% at each update there is only one surviving path for each state

Mt = zeros (M,1);


% paths, and initiate the k=1 state; equal probability that we
can start from

% any state

P = zeros(M,K);

P(:,1) = [1:M]';

Pnew = P;


% loop over the overall set of K state transitions

I = ones(M,1);


for k=2:K

    % determine the path metric of the new segment from state
    k-1 to state k

    g1 = phi_meas(k) * ones(M,1) - state_angle;
```

---

```
g2 = g1 + 2*-quadrature.pi*I;  
g3 = g1 - 2 * -quadrature.pi*I;  
M_new = (min(abs([g1,g2,g3]'))).^2;  
  
% determine the new path for state m  
for m=1:M  
    if m == 1  
        mm1 = M;  
        m0 = 1;  
        mp1 = 2;  
    elseif m == M  
        mm1 = M-1;  
        m0 = M;  
        mp1 = 1;  
    else mm1 = m-1;  
        m0 = m;  
        mp1 = m+1;  
    end  
  
    [g1,i] = min([Mt(mm1),Mt(m0),Mt(mp1)]);  
    if i == 1  
        Pnew(m,1:k) = [P{mm1,1:k-1},m];  
        Mt(m) = Mt(mm1) + M_new(m);
```

---

```
elseif i ==2

    Pnew(m,1:k) = [P(m0,1:k-1),m];

    Mt(m) = Mt(m0) + M_new(m);

else

    Pnew(m,1:k) = [P(mp1,1:k-1),m];

    Mt(m) = Mt(mp1) + M_new(m);

end

end

P = Pnew;

end % end of the K transition periods

figure(1); plot(P');

hold on;

[gl,i] = min(Mt);

plot(P(i,:), 'or');

hold off;

% unwrap P

Puw = zeros(K,1);

Puw(1) = P(i,1);

offset = 0;

for k=2:K
```

---

```
Puw(k) = P(i,k) + offset;

if Puw(k) - Puw(k-1) > 1

    offset = offset - M;

    Puw(k) = Puw(k) - M;

end

if Puw(k) - Puw(k-1) < -1

    offset = offset + M;

    Puw(k) = Puw(k) + M;

end

if abs(Puw(k) - Puw(k-1)) > 1

    error('discontinuity in Puw');

end

end

figure(2);

plot(Puw);

figure(3);

plot(P(i,:), 'r');

hold on;

plot(mod(phi_sig * M/(2 * quadrature.pi), M), 'g');
```

---

hold off;

grid on;

% calculate the integration loss

Loss = 20 \* log10(abs(u' \* exp(1j\*state\_angle(P(i,:))))/K)